

## Adv DevOps Exp 08

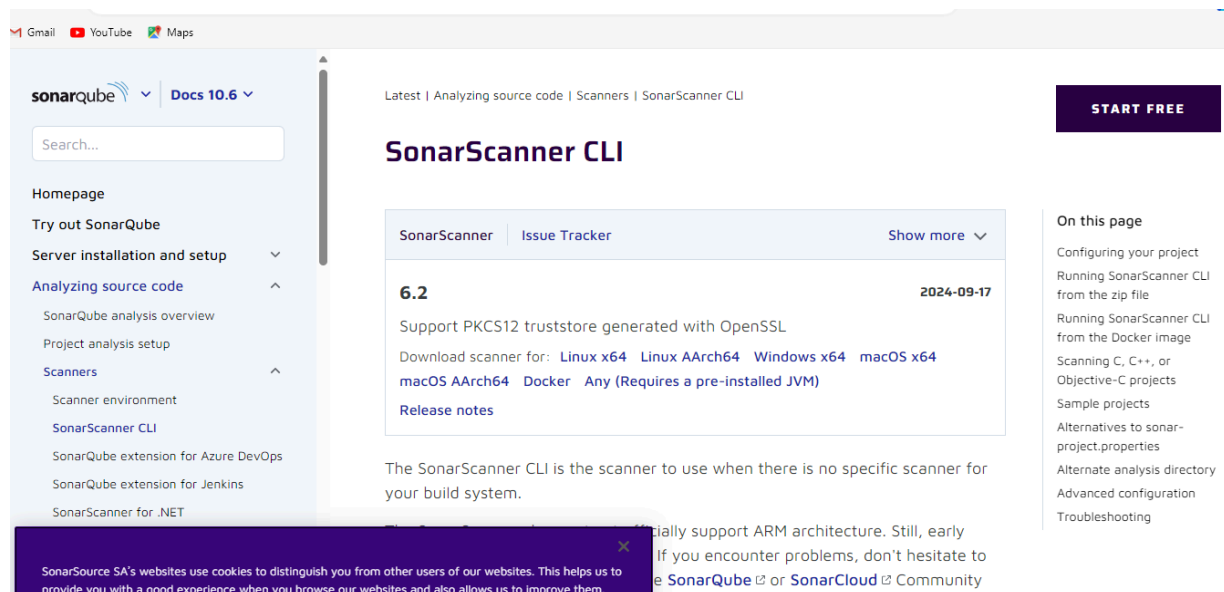
**Aim: Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web /Java / Python application.**

**(I have performed this experiment entirely on my laptop)**

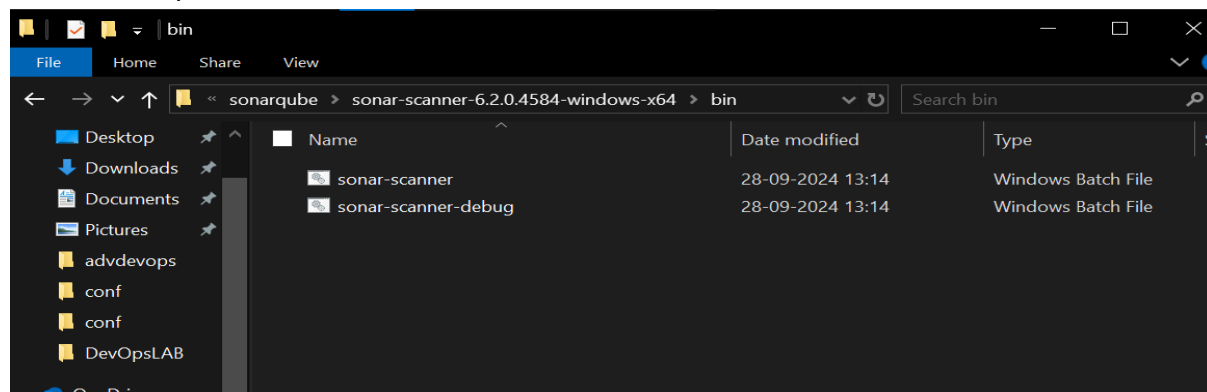
**For the Docker installation problems that i was facing during experiments 6 and 7, I resorted to the option of Resetting Windows which helped me clear the corrupted files which were causing my Laptop's Restart issues**

**Step 1:**Download sonar scanner. We will require it in the further steps while building our pipeline  
[SonarScanner CLI \(sonarsource.com\)](https://sonarsource.com)

Visit this link and download the sonarqube scanner CLI.



**Extract the zip files in a folder**

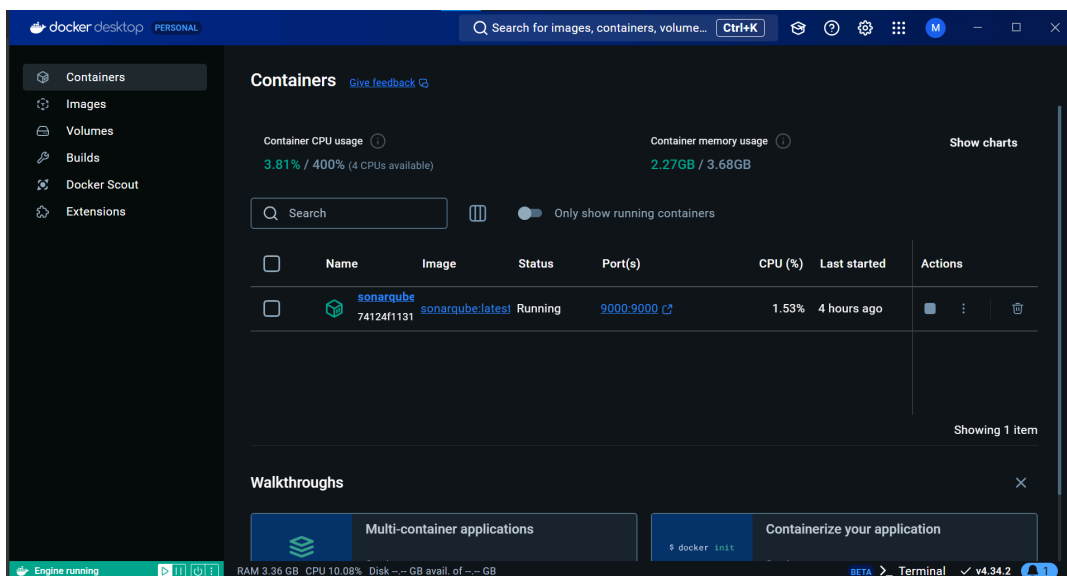


Run this command in an administrative command prompt so as to create a sonarqube image on localhost:9000

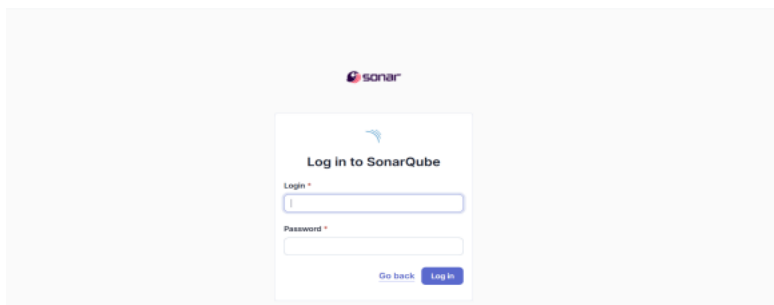
```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.19045.4957]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
4f4fb700ef54: Download complete
7d9a34308537: Download complete
bd819c9b5ead: Download complete
7478e0ac0f23: Download complete
1a5fd5c7e184: Download complete
7b87d6fa783d: Download complete
90a925ab929a: Download complete
80338217a4ab: Download complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
74124f113191d2a0857a8262e50f5023b931d5e6eb1ea2509b883831ff43b430


C:\WINDOWS\system32>
```



Visit localhost:9000 on your browser and login to your account( the default credentials are admin and admin for username and password. It allows to change it and set our own password for our account)



Next, we are supposed to Create a project. Give some suitable name to your project. A project will be generated for it automatically. We can change it or keep it the same



Projects Issues Rules Quality Profiles Quality Gates

### Create a project

**Project display name \***

Up to 255 characters. Some scanners might override the value you provide.

**Project key \***


The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '\_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.

**Main branch name \***

The name of your project's default branch [Learn More](#)

Next

## Choose global settings option for setting up your project



Projects Issues Rules Quality Profiles Quality Gates Administration More

### Set up project for Clean as You Code

The new code definition sets which part of your code will be considered new code. This helps you focus attention on the most recent changes in your project, enabling you to follow the Clean as You Code methodology. Learn more: [Defining New Code](#)

**Choose the baseline for new code for this project**

☒ Use the global setting

**Previous version**

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

☐ Define a specific setting for this project

☐ Previous version

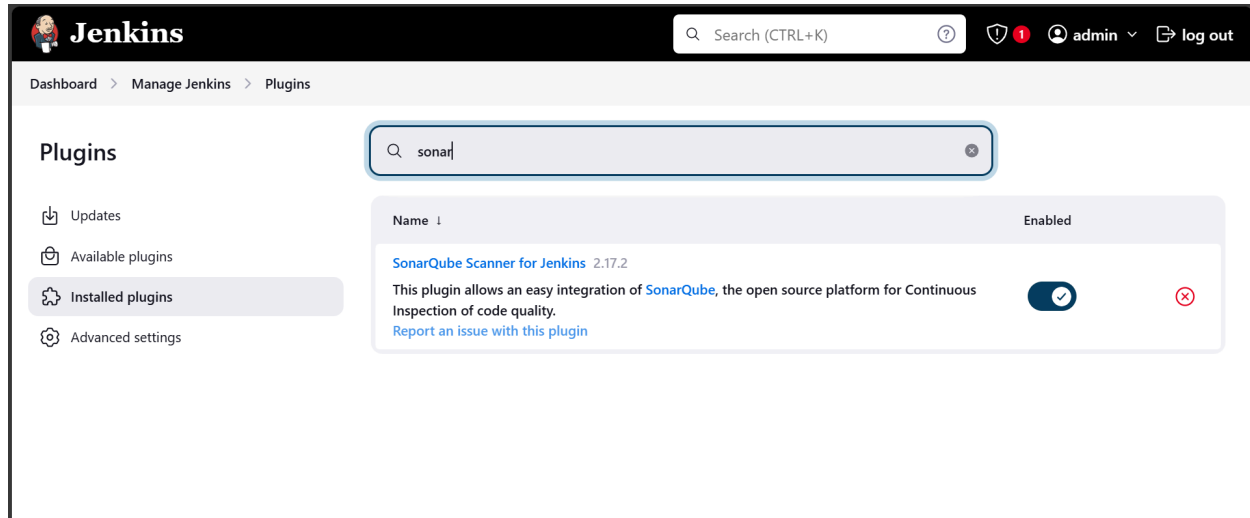
Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

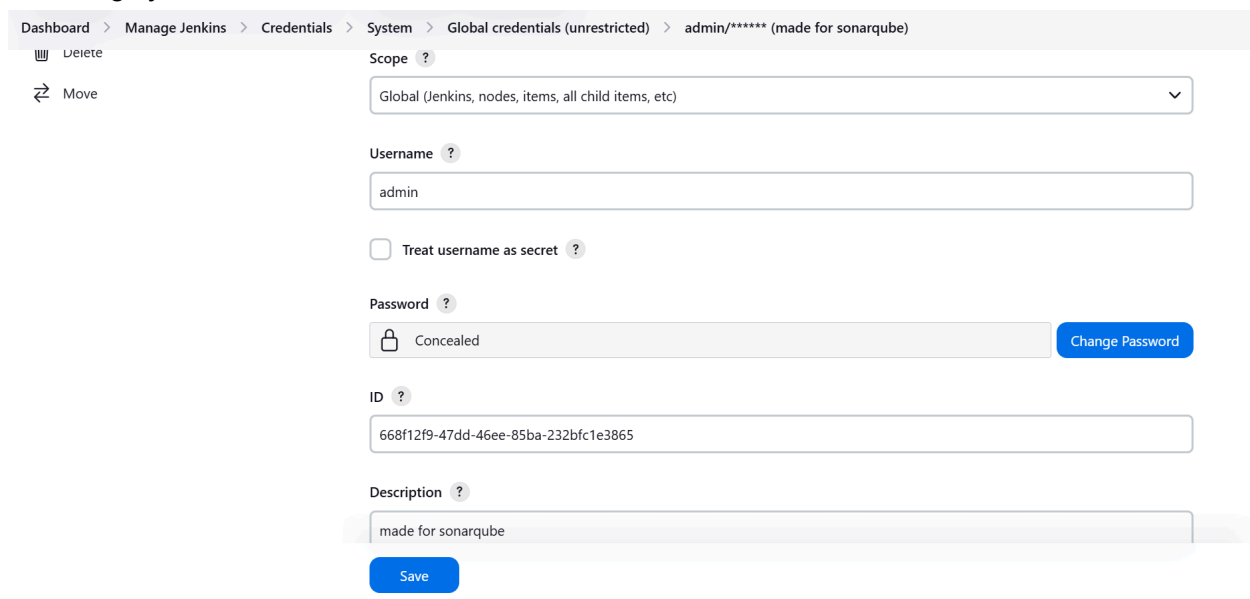
Now, on Jenkins ..

For a few contents inside the script to be built using our pipeline, we need to configure those changes inside the global configuration settings inside Manage Jenkins.

First, Go to plugins inside Manage Jenkins and install the Sonarqube Scanner plugin which is necessary for the build to be executed.



Next, we will have to add our Sonarqube project specific credentials in the Credentials section of manage jenkins



Here, i added the username i.e admin and my password ... for which jenkins generated a automated ID for the credential block.

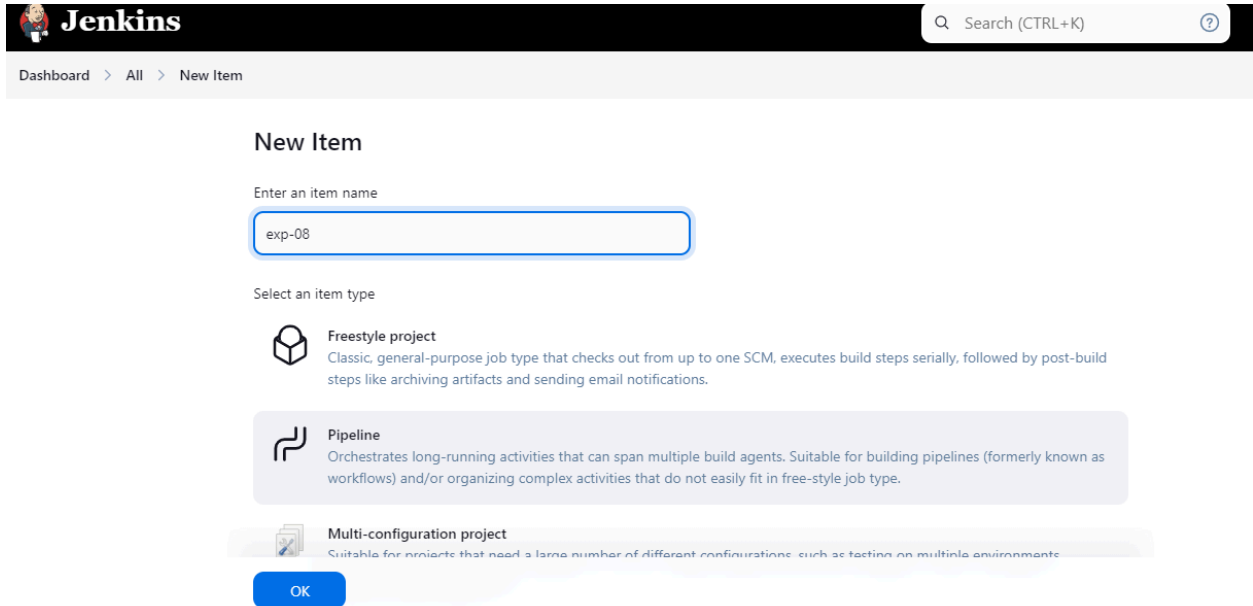
Now, our pipeline script will contain a line where we are supposed to provide the build with our source directory of the sonar-scanner batch file which we installed in the initial steps of our experiment

**C:\sonarqube\sonar-scanner-6.2.0.4584-windows-x64\bin\sonar-scanner.bat**

We will add this path inside the script to be built

Now, come to the dashboard

Create new item and select pipeline project. Give it a suitable name



**Jenkins** Search (CTRL+K) ?




Dashboard > All > New Item

### New Item

Enter an item name

exp-08

Select an item type

-  **Freestyle project**  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
-  **Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
-  **Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments.

OK

**Put this script in the Pipeline Definition section and select the mode as Pipeline script**

```
node {
  stage('Cloning the GitHub Repo') {
    git 'https://github.com/shazforiot/GOL.git'
  }

  stage('SonarQube analysis') {
    withSonarQubeEnv('sonarqube') {
      withCredentials([usernamePassword(credentialsId:
'668f12f9-47dd-46ee-85ba-232bfc1e3865', usernameVariable: 'SONAR_USER',
passwordVariable: 'SONAR_PASSWORD'))] {
        bat """
        C:/sonarqube/sonar-scanner-6.2.0.4584-windows-x64/bin/sonar-scanner.bat ^
        -D sonar.login=%SONAR_USER% ^
        -D sonar.password=%SONAR_PASSWORD% ^
        -D sonar.projectKey=exp-08 ^
        -D sonar.exclusions=vendor/**,resources/**,*/*.java ^
        -D sonar.host.url=http://127.0.0.1:9000/
        """
      }
    }
  }
}
```

Dashboard > exp-08 > Configuration

### Configure

- General
- Advanced Project Options
- Pipeline

### Pipeline

Definition

Pipeline script

```
1 node {
2   stage('Cloning the GitHub Repo') {
3     git 'https://github.com/shazforiot/GOL.git'
4   }
5
6   stage('SonarQube analysis') {
7     withSonarQubeEnv('sonarqube') {
8       withCredentials([usernamePassword(credentialsId: '668f12f9-47dd-46ee-85ba-232bfc1e3865', usernameV,
9         bat ""
10        C:/sonarqube/sonar-scanner-6.2.0.4584-windows-x64/bin/sonar-scanner.bat ^
11        -D sonar.login=${SONAR_USER} ^
12        -D sonar.password=${SONAR_PASSWORD} ^
13        -D sonar.projectKey=exp-08 ^
14        -D sonar.exclusions=vendor/**,resources/**,*/*.java ^
15        -D sonar.host.url=http://127.0.0.1:9000/
16        ""
17      )
18    }
19  }
20 }
```

☒ Use Groovy Sandbox

Save Apply

If you would notice carefully, i have used the `withCredentials` function within the pipeline script which makes the build avail the facility of using the credentials that i had set earlier

I have also added a line where i have mentioned the path of the sonar-scanner.bat file, since there are problems with system configurations with the installation of soanrqube

Now, save and build the pipeline

Jenkins

Search (CTRL+K)

Dashboard > exp-08 >

### exp-08

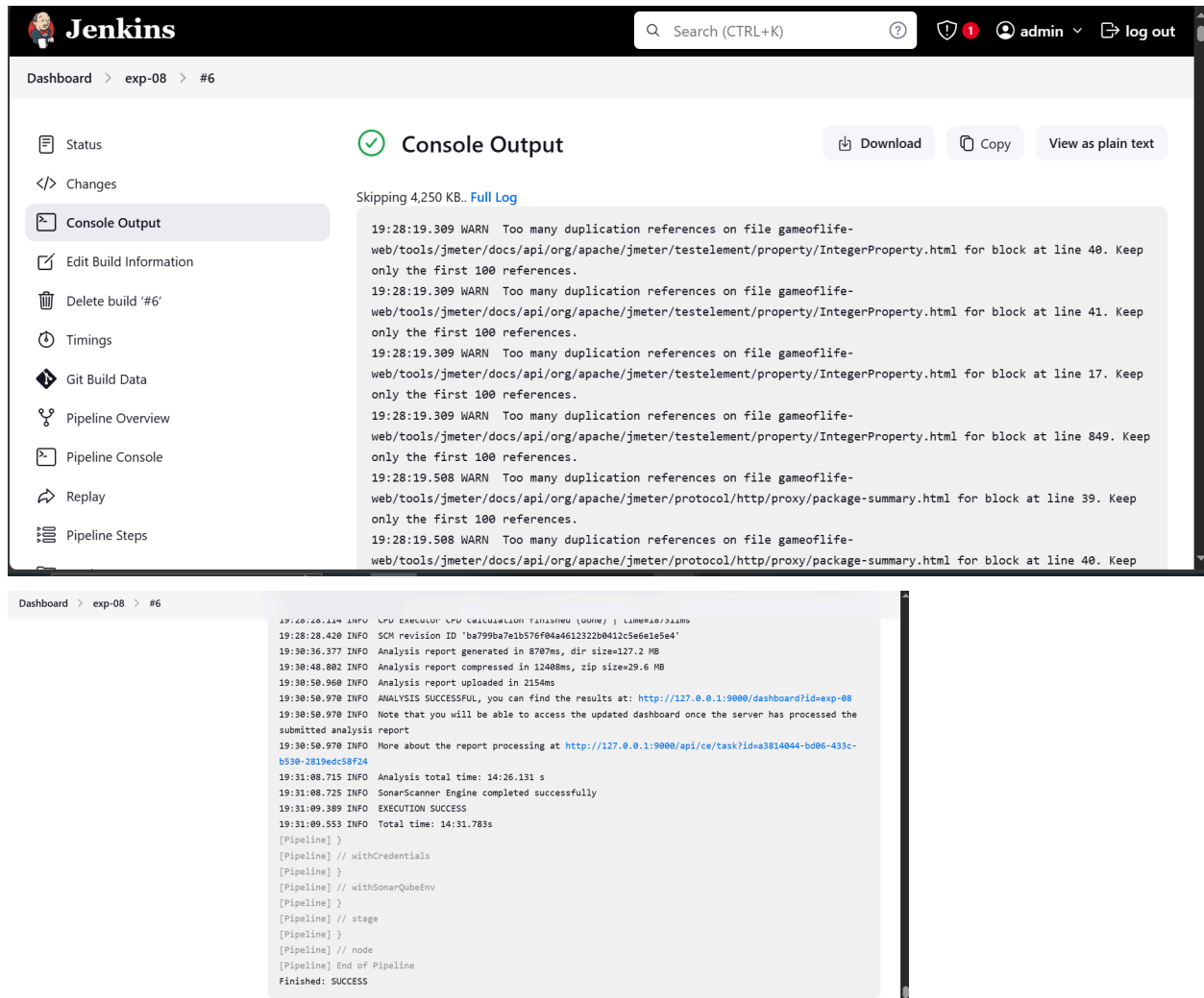
made for exp 08

### Permalinks

- Last build (#6), 1 hr 16 min ago
- Last stable build (#6), 1 hr 16 min ago
- Last successful build (#6), 1 hr 16 min ago
- Last failed build (#5), 1 hr 20 min ago
- Last unsuccessful build (#5), 1 hr 20 min ago
- Last completed build (#6), 1 hr 16 min ago

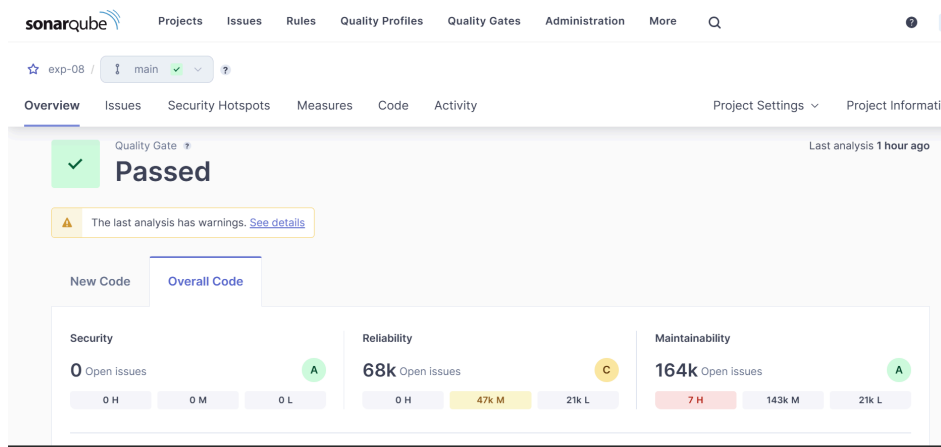
Build History trend

It generally takes 10-15 minutes for building this pipeline. We are supposed to stay patient and let the build process get completed



The screenshot shows the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and user information (admin). The main content area displays the 'Console Output' for a build named 'exp-08' at step '#6'. The output shows several warnings about 'Too many duplication references' on various files, each suggesting to keep only the first 100 references. The build status is indicated by a green checkmark. Below the console output, there are buttons for 'Download', 'Copy', and 'View as plain text'. The left sidebar contains links to various build management features like Status, Changes, Console Output, Edit Build Information, etc.

After a successful build, we are supposed to go back to our Sonarqube profile. We will see this type of a message on the screen, where it offers us multiple options to analyze our code



The screenshot shows the Sonarqube web interface. The top navigation bar includes the Sonarqube logo and links to Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. The main content area displays the 'Quality Gate' status for a project named 'exp-08'. The status is 'Passed', indicated by a green checkmark. Below the status, there is a warning message: 'The last analysis has warnings. See details'. The interface also shows a table of metrics for the project, including Security, Reliability, and Maintainability. The Security metric shows 0 Open issues, Reliability shows 68k Open issues, and Maintainability shows 164k Open issues. The table also includes a 'New Code' tab and an 'Overall Code' tab.

Within the issues section, let us explore different tabs

## 1. Bugs inside Type section

The screenshot shows the SonarQube web interface for a project named 'exp-08'. The 'Issues' tab is selected, and the 'Type' filter is set to 'Bug'. The left sidebar shows the 'Type' section with 'Bug' selected, showing 47k issues. The main area displays a list of issues, with the first issue highlighted: 'Add "lang" and/or "xml:lang" attributes to this "<html>" element'. This issue is categorized as 'Intentionality' and 'Reliability', with a severity of 'Major' and an effort of '2min'. The second issue is 'Insert a <!DOCTYPE> declaration to before this <html> tag.', categorized as 'Consistency' and 'Reliability', with a severity of 'Major' and an effort of '5min'. The top right shows a total of 46,515 issues and 14,260 effort.

## 2. Code Smell inside the Type section

The screenshot shows the SonarQube web interface for a project named 'exp-08'. The 'Issues' tab is selected, and the 'Type' filter is set to 'Code Smell'. The left sidebar shows the 'Type' section with 'Code Smell' selected, showing 164k issues. The main area displays a list of issues, with the first issue highlighted: 'Use a specific version tag for the image.'. This issue is categorized as 'Intentionality' and 'Maintainability', with a severity of 'Major' and an effort of '5min'. The second issue is 'Surround this variable with double quotes; otherwise, it can lead to unexpected behavior.', categorized as 'Intentionality' and 'Maintainability', with a severity of 'Major' and an effort of '5min'. The top right shows a total of 164,034 issues and 17,080 effort.

Embedded database should be used for evaluation purposes only

## 3. Consistency inside Clean Code Attribute



The screenshot shows the SonarQube 'Issues' tab for project 'exp-08'. The left sidebar displays a list of quality attributes: 'Clean Code Attribute' (1 issue), 'Consistency' (164k), 'Intentionality' (268), 'Adaptability' (0), 'Responsibility' (0), and 'Software Quality'. The main panel shows a list of issues. The first issue is 'Remove this deprecated "width" attribute.' with a 'Maintainability' quality gate. The second issue is 'Remove this deprecated "align" attribute.' with a 'Maintainability' quality gate. The top right of the main panel shows '163,766 issues' and '1705d effort'.

#### 4. Reliability inside Software Quality

The screenshot shows the SonarQube 'Issues' tab for project 'exp-08' with 'Reliability' selected in the left sidebar. The main panel shows a list of issues. The first issue is 'Anchors must have content and the content must be accessible by a screen reader.' with a 'Reliability' quality gate. The second issue is 'Anchors must have content and the content must be accessible by a screen reader.' with a 'Reliability' quality gate. The top right of the main panel shows '20,856 issues' and '217d effort'.

## 5. Maintainability

The screenshot displays the SonarQube web interface for a project named 'exp-08'. The 'Issues' tab is active, showing a list of issues. The left sidebar shows a list of quality categories: Responsibility (0), Software Quality (1), Security (0), Reliability (21k), and Maintainability (164k). The main area displays a list of issues. Two issues are highlighted with a blue box: 'Remove this deprecated "width" attribute.' and 'Remove this deprecated "align" attribute.'. Both issues are categorized as 'Maintainability' and 'Consistency'. The top of the interface shows the SonarQube logo, navigation tabs (Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More), and a search bar. The bottom of the interface has a yellow banner stating 'Embedded database should be used for evaluation purposes only'.

### Conclusion:

In this experiment, we set up a Jenkins CI/CD pipeline integrated with SonarQube to automate static analysis on a sample application. Jenkins was configured to trigger builds and run SonarQube's analysis with every code change, detecting bugs, code smells, and security vulnerabilities. This pipeline provided continuous monitoring and ensured early detection of issues, improving code quality and security. The experiment showcased how integrating CI/CD pipelines with SonarQube enhances development efficiency and ensures better, more reliable software.